Data Science Essentials

Topic: Analysis of Missing Values

Category: Data Visualization

Created By: Mohammed Misbahullah Sheriff

- LinkedIn
- GitHub

1. Importing Libraries

In [1]:
```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import missingno as msno

import os
```

2. Getting the Data

In [2]:
```
path = os.path.join("C:\Python Programs\datasets", "house_price_dataset.csv")

df = pd.read_csv(path)
print("Data Shape", df.shape)
df.head()
```

```
Data Shape (1460, 81)
```

Out[2]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lv
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lv
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lv
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lv
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lv

5 rows × 81 columns

In [3]:
```
X = df.drop(columns="SalePrice")
y = df.SalePrice.copy()

print(X.shape, y.shape)
```

```
(1460, 80) (1460,)
```

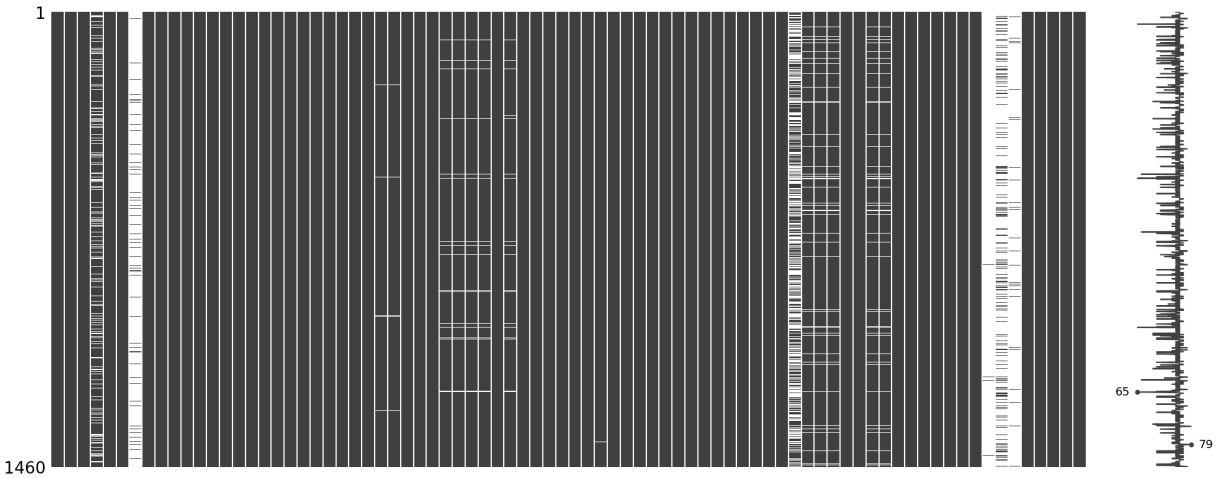
- The dataset has 1,460 observations and 80 features

3. Demo 1 - missingno

3.1 Matrix

- This will indicate the rows that contain missing values
- In this plot, darkness indicates availability of data
- Thus, lighter the bar, the more missing values a feature has

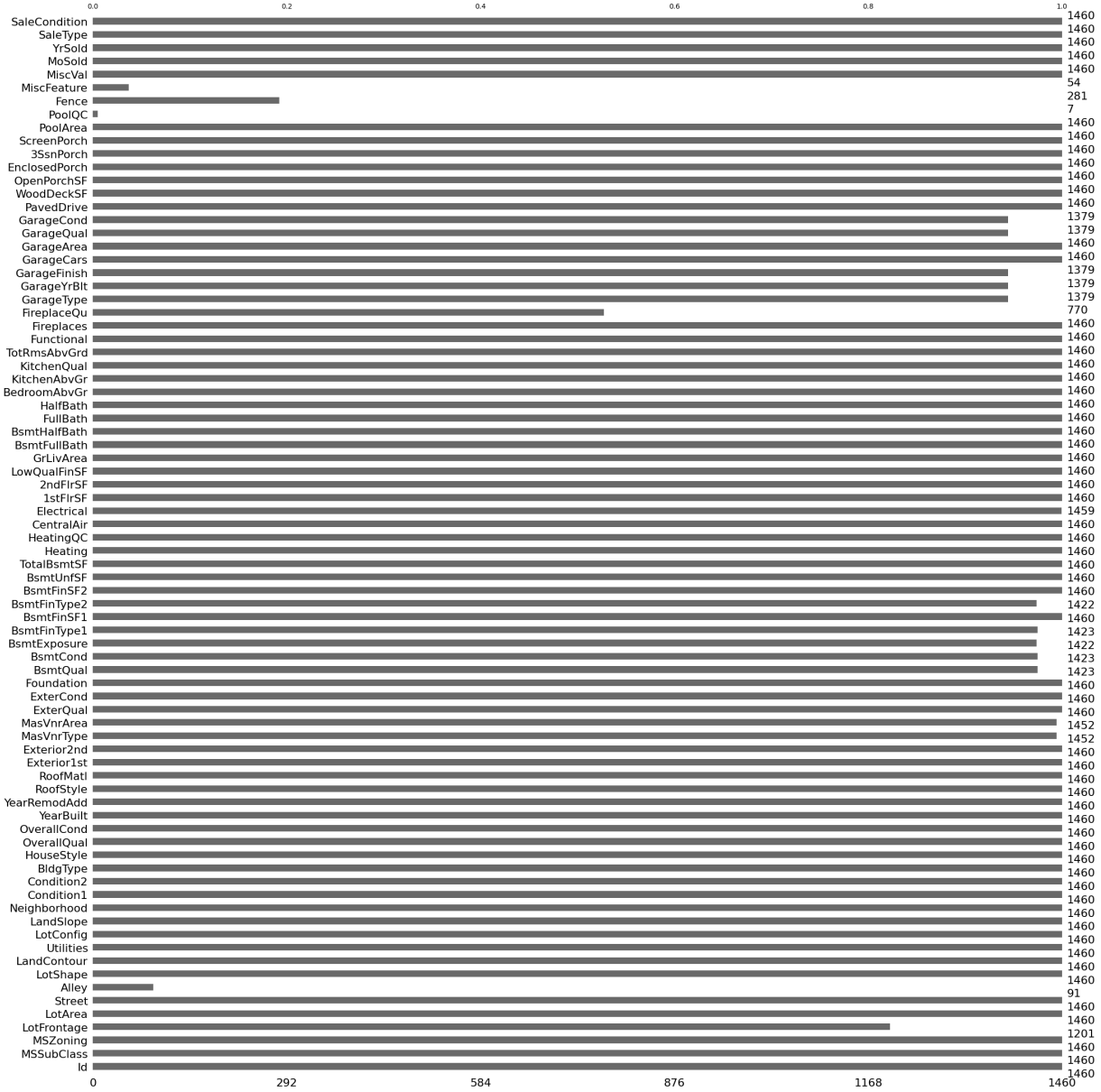
In [4]:
```
msno.matrix(X)
plt.show()
```



3.2 - Bar Plot

- This will plot bars for each feature
- Length of the bar indicates availability of data
- Thus, shorter the bar, the more missing values a feature has

In [5]:
```
msno.bar(X)
plt.show()
```



4. Demo 2 - Custom Functions

In [6]:
```
def missing_info(data):
  """
  Description:
  ------------
  This function will accept a dataframe and return a dataframe describing info abou
```

```
    The output dataframe will show the count and percentage of missing values in each
    with the feature name as index.
    The percentages are with respect to the total no. of observations in the given da

    Parameters:
    -----------
    data: dataframe
          Input dataframe
    """
    na_cols = [col for col in data.columns if data[col].isna().any()]
    na_frequencies = [data[col].isna().sum() for col in na_cols]
    na_percentages = [data[col].isna().mean() for col in na_cols]

    return (
        pd
        .DataFrame(data={
            "column": na_cols,
            "count": na_frequencies,
            "percentage": na_percentages
        })
        .set_index("column")
        .sort_values("count", ascending=False)
    )
```

In [7]:

```
def plot_missing_info(data,
                      figsize=(6, 4),
                      color="#1745e8",
                      show_bar_labels=True):
    """
    Description:
    ------------
    This function will accept a dataframe and return a bar plot showing the cou
    in each feature in descending order.

    Parameters:
    -----------
    data: dataframe
          Input dataframe

    figsize = tuple -> (width, height)
              The dimensions of the bar plot figure

    color = str
            Color to use for the bars. Any valid color string will be accepted.

    show_bar_labels: bool
                     Whether to display the count of missing values for each fe
    """
    fig, ax = plt.subplots(figsize=figsize)

    bar = (
        missing_info(data)
        .loc[:, "count"]
        .plot
        .bar(
            color=color,
```

```
                ax=ax,
                alpha=0.7,
                edgecolor="black"
            )
        )

        ax.set_xlabel("Feature", fontweight="bold", fontsize=11)
        ax.set_ylabel("Count", fontweight="bold", fontsize=11)
        ax.set_title("Missing Values Counts for each Feature", fontweight="bold", f

        ax.set_xticklabels(
            ax.get_xticklabels(),
            rotation=45,
            ha="right"
        )

        if show_bar_labels:
                containers = bar.containers[0]
                labels = [f"{count:,}" for count in containers.datavalues]
                ax.bar_label(
                        containers,
                        labels=labels,
                        padding=2
                )

        plt.show()
```

4.1 Calling `missing_info()`

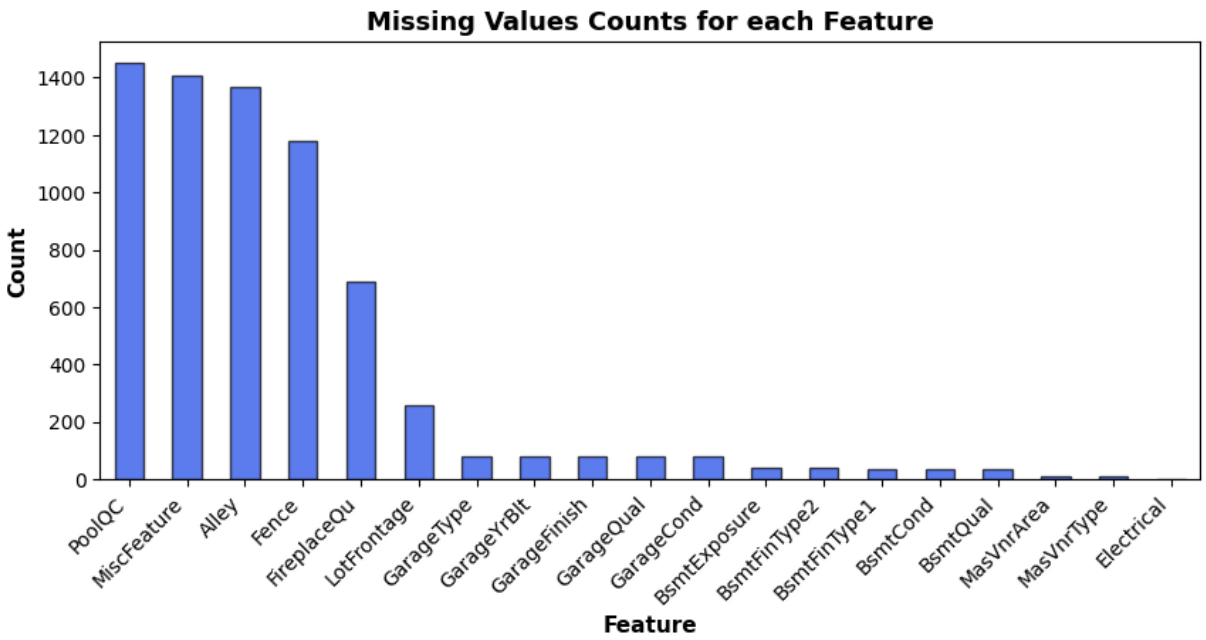
In [8]:
```
missing_info(X)
```

Out[8]:

column	count	percentage
PoolQC	1453	0.995205
MiscFeature	1406	0.963014
Alley	1369	0.937671
Fence	1179	0.807534
FireplaceQu	690	0.472603
LotFrontage	259	0.177397
GarageType	81	0.055479
GarageYrBlt	81	0.055479
GarageFinish	81	0.055479
GarageQual	81	0.055479
GarageCond	81	0.055479
BsmtExposure	38	0.026027
BsmtFinType2	38	0.026027
BsmtFinType1	37	0.025342
BsmtCond	37	0.025342
BsmtQual	37	0.025342
MasVnrArea	8	0.005479
MasVnrType	8	0.005479
Electrical	1	0.000685

4.2 Calling `plot_missing_info()`

In [9]:
```
plot_missing_info(
    X,
    figsize=(10, 4),
    show_bar_labels=False
)
```



In [10]:
```
plot_missing_info(
    X,
    figsize=(10, 6),
    color="green",
    show_bar_labels=True
)
```